Another_Association_Mining

November 12, 2025

[24]:
```
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules

# Load the dataset
data = pd.read_csv('basket_analysis.csv')

# Drop unnecessary columns (like indexes)
data_clean = data.drop(columns=['Unnamed: 0', 'next'], errors='ignore')

# Preview the data
data_clean.head()
```

[24]:

	Apple	Bread	Butter	Cheese	Corn	Dill	Eggs	Ice cream	Kidney Beans	Milk	Nutmeg	Onion	Sugar	Unicorn	Yogurt	chocolate
0	False	True	False	False	True	True	False	True	False	False	False	False	True	False	True	True
1	False	False	False	False	False	False	False	False	False	True	False	False	False	False	False	False
2	True	False	True	False	False	True	False	True	False	True	False	False	False	False	True	True
3	False	False	True	True	False	True	False	False	False	True	True	True	False	False	False	False
4	True	True	False	False	False	False	False	False	False	False	False	False	False	False	False	False

[8]:
```
# Case a: min_support = 0.5, min_confidence = 0.75
frequent_itemsets_a = apriori(data_clean, min_support=0.5, use_colnames=True)

# Print number of frequent itemsets
print("Case a - Number of frequent itemsets:", len(frequent_itemsets_a))

# Generate association rules (if any)
if len(frequent_itemsets_a) > 0:
    rules_a = association_rules(frequent_itemsets_a, metric="confidence",␣
 ↪min_threshold=0.75)
else:
```

```
    rules_a = pd.
 ↪DataFrame(columns=['antecedents','consequents','support','confidence','lift'])

# Print number of rules and the rules themselves
print("Case a - Number of rules:", len(rules_a))
print(rules_a)
```

```
Case a - Number of frequent itemsets: 0
Case a - Number of rules: 0
Empty DataFrame
Columns: [antecedents, consequents, support, confidence, lift]
Index: []
```

[9]:
```
# Case b: min_support = 0.6, min_confidence = 0.6
frequent_itemsets_b = apriori(data_clean, min_support=0.6, use_colnames=True)

# Print number of frequent itemsets
print("Case b - Number of frequent itemsets:", len(frequent_itemsets_b))

# Generate association rules (if any)
if len(frequent_itemsets_b) > 0:
    rules_b = association_rules(frequent_itemsets_b, metric="confidence",␣
 ↪min_threshold=0.6)
else:
    rules_b = pd.
 ↪DataFrame(columns=['antecedents','consequents','support','confidence','lift'])

# Print number of rules and the rules themselves
print("Case b - Number of rules:", len(rules_b))
print(rules_b)
```

```
Case b - Number of frequent itemsets: 0
Case b - Number of rules: 0
Empty DataFrame
Columns: [antecedents, consequents, support, confidence, lift]
Index: []
```

[10]:
```
# Case C: min_support = 0.3, min_confidence = 0.6
frequent_itemsets_c = apriori(data_clean, min_support=0.3, use_colnames=True)

print("Case C - Number of frequent itemsets:", len(frequent_itemsets_c))

if len(frequent_itemsets_c) > 0:
    rules_c = association_rules(frequent_itemsets_c, metric="confidence",␣
 ↪min_threshold=0.6)
else:
    rules_c = pd.
 ↪DataFrame(columns=['antecedents','consequents','support','confidence','lift'])
```

```
print("Case C - Number of rules:", len(rules_c))
print(rules_c)
```

```
Case C - Number of frequent itemsets: 16
Case C - Number of rules: 0
Empty DataFrame
Columns: [antecedents, consequents, antecedent support, consequent support,
support, confidence, lift, representativity, leverage, conviction,
zhangs_metric, jaccard, certainty, kulczynski]
Index: []
```

0.1 association_rules()

- It generates rules A → B from frequent itemsets only if the itemset has at least 2 items.
- association_rules() cannot create a rule from a single item, because you need an antecedent and a consequent.

[11]:
```
# Case D: min_support = 0.3, min_confidence = 0.4
frequent_itemsets_d = apriori(data_clean, min_support=0.3, use_colnames=True)

print("Case D - Number of frequent itemsets:", len(frequent_itemsets_d))
print("Frequent itemsets:\n", frequent_itemsets_d)

# Generate association rules
if len(frequent_itemsets_d) > 0:
    rules_d = association_rules(frequent_itemsets_d, metric="confidence",␣
 ↪min_threshold=0.4)
else:
    rules_d = pd.
 ↪DataFrame(columns=['antecedents','consequents','support','confidence','lift'])

print("\nCase D - Number of rules:", len(rules_d))
print("Association rules:\n", rules_d)
```

```
Case D - Number of frequent itemsets: 16
Frequent itemsets:
      support        itemsets
0   0.383383         (Apple)
1   0.384384         (Bread)
2   0.420420        (Butter)
3   0.404404        (Cheese)
4   0.407407          (Corn)
5   0.398398          (Dill)
6   0.384384          (Eggs)
7   0.410410     (Ice cream)
8   0.408408  (Kidney Beans)
9   0.405405          (Milk)
```

```
10  0.401401        (Nutmeg)
11  0.403403         (Onion)
12  0.409409         (Sugar)
13  0.389389       (Unicorn)
14  0.420420         (Yogurt)
15  0.421421     (chocolate)

Case D - Number of rules: 0
Association rules:
 Empty DataFrame
Columns: [antecedents, consequents, antecedent support, consequent support,
support, confidence, lift, representativity, leverage, conviction,
zhangs_metric, jaccard, certainty, kulczynski]
Index: []
```

[14]:
```
# Case E: min_support = 0.2, min_confidence = 0.5
frequent_itemsets_e = apriori(data_clean, min_support=0.2, use_colnames=True)

# Print number of frequent itemsets
print("Case E - Number of frequent itemsets:", len(frequent_itemsets_e))

# Generate association rules if frequent itemsets exist
if len(frequent_itemsets_e) > 0:
    rules_e = association_rules(frequent_itemsets_e, metric="confidence",␣
 ↪min_threshold=0.5)
    print("Case E - Number of rules:", len(rules_e))
    print("Association rules:\n", rules_e)
else:
    rules_e = pd.
 ↪DataFrame(columns=['antecedents','consequents','support','confidence','lift'])
    print("Case E - Number of rules: 0")
    print("Association rules:\n", rules_e)
```

```
Case E - Number of frequent itemsets: 22
Case E - Number of rules: 3
Association rules:
    antecedents  consequents  antecedent support  consequent support   support
\
0  (Ice cream)     (Butter)            0.410410            0.420420  0.207207
1       (Milk)  (chocolate)            0.405405            0.421421  0.211211
2  (chocolate)       (Milk)            0.421421            0.405405  0.211211

   confidence      lift  representativity  leverage  conviction  \
0    0.504878  1.200889               1.0  0.034662    1.170579
1    0.520988  1.236263               1.0  0.040365    1.207857
2    0.501188  1.236263               1.0  0.040365    1.192021

   zhangs_metric   jaccard  certainty  kulczynski
```

```
0       0.283728  0.332263    0.145722     0.498868
1       0.321413  0.343089    0.172088     0.511088
2       0.330310  0.343089    0.161088     0.511088
```

Focus on support, confidence and lift columns.

1. Ice cream → Butter

- Appears together in ~20.7% of transactions.
- If a customer buys Ice cream, there's a 50.5% chance they also buy Butter.
- Lift = 1.201 → positive association (buying Ice cream slightly increases the likelihood of buying Butter).

2. Milk → chocolate

- Appears together in ~21.1% of transactions.
- If a customer buys Milk, there's a 52.1% chance they also buy chocolate.
- Lift = 1.236 → positive association (buying Milk increases the likelihood of buying chocolate).

3. chocolate → Milk

- Appears together in ~21.1% of transactions.
- If a customer buys chocolate, there's a 50.1% chance they also buy Milk.
- Lift = 1.236 → positive association (buying chocolate increases the likelihood of buying Milk).

[17]:
```
# Case F: min_support = 0.2, min_confidence = 0.3 (lower confidence)
frequent_itemsets_f = apriori(data_clean, min_support=0.2, use_colnames=True)

print("Case F - Number of frequent itemsets:", len(frequent_itemsets_f))

# Generate association rules with lower confidence to get more rules
if len(frequent_itemsets_f) > 0:
    rules_f = association_rules(frequent_itemsets_f, metric="confidence",␣
 ↪min_threshold=0.3)
    print("Case F - Number of rules:", len(rules_f))
    print("Association rules:\n", rules_f)
else:
    rules_f = pd.
 ↪DataFrame(columns=['antecedents','consequents','support','confidence','lift'])
    print("Case F - Number of rules: 0")
    print("Association rules:\n", rules_f)
```

```
Case F - Number of frequent itemsets: 22
Case F - Number of rules: 12
Association rules:
        antecedents     consequents  antecedent support  consequent support  \
0         (Butter)     (Ice cream)            0.420420            0.410410
1      (Ice cream)        (Butter)            0.410410            0.420420
2   (Kidney Beans)        (Butter)            0.408408            0.420420
3         (Butter)  (Kidney Beans)            0.420420            0.408408
4         (Butter)     (chocolate)            0.420420            0.421421
```

```
5      (chocolate)        (Butter)            0.421421            0.420420
6   (Kidney Beans)        (Cheese)            0.408408            0.404404
7         (Cheese)  (Kidney Beans)            0.404404            0.408408
8      (Ice cream)     (chocolate)            0.410410            0.421421
9      (chocolate)     (Ice cream)            0.421421            0.410410
10          (Milk)     (chocolate)            0.405405            0.421421
11     (chocolate)          (Milk)            0.421421            0.405405

     support  confidence      lift  representativity  leverage  conviction  \
0   0.207207    0.492857  1.200889               1.0  0.034662    1.162571
1   0.207207    0.504878  1.200889               1.0  0.034662    1.170579
2   0.202202    0.495098  1.177626               1.0  0.030499    1.147905
3   0.202202    0.480952  1.177626               1.0  0.030499    1.139764
4   0.202202    0.480952  1.141262               1.0  0.025028    1.114693
5   0.202202    0.479810  1.141262               1.0  0.025028    1.114169
6   0.200200    0.490196  1.212143               1.0  0.035038    1.168284
7   0.200200    0.495050  1.212143               1.0  0.035038    1.171583
8   0.202202    0.492683  1.169098               1.0  0.029246    1.140467
9   0.202202    0.479810  1.169098               1.0  0.029246    1.133412
10  0.211211    0.520988  1.236263               1.0  0.040365    1.207857
11  0.211211    0.501188  1.236263               1.0  0.040365    1.192021

    zhangs_metric   jaccard  certainty  kulczynski
0        0.288629  0.332263   0.139837    0.498868
1        0.283728  0.332263   0.145722    0.498868
2        0.254963  0.322684   0.128848    0.488025
3        0.260247  0.322684   0.122625    0.488025
4        0.213564  0.316119   0.102892    0.480381
5        0.213933  0.316119   0.102470    0.480381
6        0.295838  0.326797   0.144043    0.492623
7        0.293849  0.326797   0.146454    0.492623
8        0.245323  0.321145   0.123167    0.486246
9        0.249991  0.321145   0.117708    0.486246
10       0.321413  0.343089   0.172088    0.511088
11       0.330310  0.343089   0.161088    0.511088
```

[23]:
```python
# Prepare a clean table
rules_table = rules_f.copy()

# Round support, confidence, lift to 3 decimals
rules_table['support'] = rules_table['support'].round(3)
rules_table['confidence'] = rules_table['confidence'].round(3)
rules_table['lift'] = rules_table['lift'].round(3)

# Keep original order
rules_table = rules_table[['antecedents', 'consequents', 'support',␣
 ↪'confidence', 'lift']]
```

```
rules_table['antecedents'] = rules_table['antecedents'].apply(lambda x: ', '.
  ↪join(list(x)))
rules_table['consequents'] = rules_table['consequents'].apply(lambda x: ', '.
  ↪join(list(x)))
print(rules_table)
```

```
     antecedents   consequents  support  confidence   lift
0         Butter     Ice cream    0.207       0.493  1.201
1      Ice cream        Butter    0.207       0.505  1.201
2   Kidney Beans        Butter    0.202       0.495  1.178
3         Butter  Kidney Beans    0.202       0.481  1.178
4         Butter     chocolate    0.202       0.481  1.141
5      chocolate        Butter    0.202       0.480  1.141
6   Kidney Beans        Cheese    0.200       0.490  1.212
7         Cheese  Kidney Beans    0.200       0.495  1.212
8      Ice cream     chocolate    0.202       0.493  1.169
9      chocolate     Ice cream    0.202       0.480  1.169
10          Milk     chocolate    0.211       0.521  1.236
11     chocolate          Milk    0.211       0.501  1.236
```

1. **Function**
   - `apyori`: `apriori(records, min_support=..., min_confidence=..., min_lift=..., min_length=...)`
   - `mlxtend`: `apriori(df, min_support=..., use_colnames=True)`
2. **Parameters**
   - `apyori`: Accepts `min_support`, `min_confidence`, `min_lift`, `min_length` directly.
   - `mlxtend`: Only `min_support` and `use_colnames` in `apriori()`; confidence, lift, etc., are applied separately via `association_rules()`.
3. **Input Type**
   - `apyori`: List of lists (e.g., `[['Milk','Bread'], ['Eggs','Milk']]`).
   - `mlxtend`: Pandas DataFrame of boolean type (e.g., `basket`).
4. **Output**
   - `apyori`: Generator of `RelationRecord` objects containing itemsets, support, confidence, and lift.
   - `mlxtend`: DataFrame of frequent itemsets with `support`; rules generated separately.
5. **Rule Generation**
   - `apyori`: Rules are generated automatically as part of the output.
   - `mlxtend`: Rules are generated separately using `association_rules(frequent_itemsets)`.
6. **Speed / Dataset Size**
   - `apyori`: Slow on large datasets; suitable for small datasets for learning purposes.
   - `mlxtend`: Optimized for moderate to large datasets; efficient with pandas DataFrames.
7. **Ease of Use**

- `apyori`: Simple, all-in-one for small educational datasets.
- `mlxtend`: Slightly more steps, but more flexible and scalable for analysis.

0.1.1 Conclusion

1. **Frequent Itemsets**
   - Using the Apriori algorithm, the number of frequent itemsets varied depending on the **support threshold**:
     - **High support (50–60%)** → very few frequent itemsets, mostly single items.
     - **Moderate support (30–40%)** → still sparse, often resulting in 0 association rules.
     - **Lower support (20%)** → 22 frequent itemsets were found, including multi-item sets.
2. **Association Rules**
   - The number of rules depends on both **support** and **confidence thresholds**:
     - **High confidence (0.5) with 20% support** → 3 rules generated (Case E).
     - **Lowering confidence to 0.3–0.4** → 12 rules generated (Case F), showing more associations.
   - Most rules involved commonly bought items together:
     - **Milk chocolate**, **Ice cream Butter**, **Cheese Kidney Beans**, etc.
3. **Rule Metrics**
   - **Support**: 20–21% for most rules — these itemsets appear in about one-fifth of transactions.
   - **Confidence**: 49–52% — moderate probability of consequent given antecedent.
   - **Lift**: 1.17–1.24 — positive association, meaning buying the antecedent slightly increases the likelihood of buying the consequent.
4. **Observations**
   - **Sparse dataset**: Many single items appear frequently, but combinations are rarer.
   - **Lowering confidence** while keeping 20% support helped generate more rules without changing the support threshold.
   - Some rules were symmetric (A → B and B → A) — both have similar confidence and lift.
5. **Conclusion**
   - The dataset demonstrates that **frequent grocery item combinations can be identified**, but thresholds need careful adjustment.
   - **20% support** and **moderate confidence (0.3–0.5)** produced the most actionable rules.
   - Strongest associations (higher lift) include **Milk chocolate** and **Ice cream Butter**, which could inform promotions or recommendations.

[ ]: